Efficient Algorithms for Graph Optimization Problems

(Hatékony algoritmusok gráfoptimalizálási feladatokra)

Doktori értekezés tézisei

KOVÁCS PÉTER

Témavezető: Király Zoltán, Ph.D.

Eötvös Loránd Tudományegyetem
Informatikai Kar

Informatika Doktori Iskola
Iskolavezető: Csuhaj Varjú Erzsébet, D.Sc.

Doktori program: Az Informatika Alapjai és Módszertana
Programvezető: Horváth Zoltán, Ph.D.

Budapest, 2019

1. Bevezetés

A gráfok számos tudományterületen a modellezés nélkülözhetetlen eszközeivé váltak, elsősorban kifejezőerejüknek és a hozzájuk kapcsolódó mély elméleti háttérnek köszönhetően. Az elmúlt néhány évtizedben intenzív kutatómunka irányult a gráfokhoz kapcsolódó kombinatorikus optimalizálási algoritmusok fejlesztésére és elemzésére.

Az értekezésben két feladatot vizsgálunk: a *minimális költségű folyam* és a *legnagyobb közös részgráf* problémát. Mindkettő széles körben alkalmazott alapvető optimalizálási feladat, kiterjedt irodalommal. A kutatás legfontosabb eredményét különböző megoldási módszerek hatékony megvalósítása jelenti, beleértve új heurisztikus javítások kifejlesztését, valamint gráfok és fák speciális reprezentációját. Alapos tapasztalati elemzéseket is végeztünk, hogy összehasonlítsuk az implementált algoritmusokat más elérhető megoldóprogramokkal, nyílt forrású és kereskedelmi szoftverekkel egyaránt. Ezek a mérések igazolták, hogy a szerző által adott leghatékonyabb algoritmusok az esetek többségében gyorsabbak, illetve jobb eredményeket adnak, mint más implementációk.

A bemutatott munka az [1, 2, 3, 4] folyóiratcikkeken és az [5, 6] konferenciacikkeken alapul. Ezekre a publikációkra összesen több mint 300 független hivatkozás történt a Google Scholar [a] szerint, amelyek közül több mint 220 ellenőrzött idézésként szerepel az MTMT (Magyar Tudományos Művek Tára) [b] adatbázisában is.

2. Minimális költségű folyamok

A *minimális költségű folyam* (*minimum-cost flow*, MCF) feladatban egy minimális összköltségű szállítási tervet kell meghatároznunk, amely adott mennyiségű árucikket vagy más entitást eljuttat egy irányított hálózat termelő csúcsaiból a fogyasztó csúcsaiba. A hálózat éleihez kapacitáskorlátokat és költségértékeket rendelünk. Ennek a modellnek rendkívül szerteágazó alkalmazási területei vannak, például szállítmányozás, kommunikáció, hálózattervezés, kémiai informatika, ütemezés, gyártási folyamatok optimalizálása, erőforrás-elosztás és evakuálási tervek készítése [7, 18].

[a] https://scholar.google.com/citations?user=7yee1R0AAAAJ
[b] https://m2.mtmt.hu/gui2/?type=authors&mode=browse&sel=10033546

2.1. Implementált algoritmusok

Az MCF probléma megoldására a szerző hét különböző algoritmust valósított meg [2, 3, 6]. Ezek az algoritmusok ismert módszereken alapulnak, de az implementáció során az alábbiakban felsorolt javításokat és új ötleteket is alkalmaztuk.

– **Simple cycle-canceling (SCC)**: egy egyszerű primál algoritmus, amely minden iterációban megkeres és megszüntet egy negatív költségű irányított kört a reziduális hálózatban. Egy tetszőleges megengedett megoldásból indulva ezek a műveletek folyamatosan csökkentik a folyam összköltségét, amíg egy optimális megoldást nem találunk.

  Hozzájárulás:

  - Negatív körök keresésére a Bellman–Ford-algoritmust alkalmazzuk, az iterációk számára vonatkozó korlát fokozatos növelésével, ami általában sokkal gyorsabbá teszi a keresést.

– **Minimum-mean cycle-canceling (MMCC):** egy jól ismert erősen polinomiális algoritmus, amely minden iterációban a reziduális hálózat egy minimális átlagköltségű körét szünteti meg.

  Hozzájárulás:

  - Minimális átlagú körök keresésére egy olyan kombinált algoritmust alkalmazunk, amely garantálja az erősen polinomiális futási időt és a gyakorlatban is kiemelkedően hatékony.

– **Cancel-and-tighten (CAT):** az MMCC algoritmus egy javított változata, amely a duál megoldás (csúcspotenciálok) felhasználásával gyorsabban talál negatív köröket.

  Hozzájárulás:

  - Gyakrabban végezzük el az úgynevezett *tighten* lépés szigorúbb változatát, vagyis egy minimális átlagú kör keresését: $k = n$ helyett minden $k = \lfloor \sqrt{n} \rfloor$ iteráció után (ahol n a gráf csúcsainak száma).
  - Minimális átlagú körök keresésére ugyanazt a kombinált algoritmust alkalmazzuk, mint az MMCC algoritmusban.

– **Successive shortest path (SSP):** egy alapvető duál módszer, amely a reziduális hálózat legrövidebb útjai mentén küld folyamot a termelő csúcsokból a fogyasztó csúcsokba egészen addig, amíg az összes termelést el nem szállítja.

Hozzájárulás:

- A reziduális hálózat reprezentálására egy hatékony tárolási technikát alkalmazunk, amely lehető teszi egy csúcs kimenő éleinek gyors bejárását.

– **Capacity-scaling (CAS):** az SSP algoritmus egy javított változata, amely kapacitásskálázó fázisok végrehajtásával biztosítja, hogy kellően nagy mennyiségű folyamot küldjünk a legrövidebb utak mentén.

Hozzájárulás:

- A reziduális hálózatot úgy reprezentáljuk, ahogy az SSP algoritmusban.
- A skálázási faktort 2 helyett 4-re állítjuk.

– **Cost-scaling (COS):** egy primál-duál algoritmus, amely egy költségskálázó technikát alkalmazva lokális *push* (*pumpálás*) és *relabel* (*átcímkézés*) műveleteket hajt végre. Ez a módszer a maximális folyam feladatot megoldó *előfolyam* algoritmus egy általánosítása.

Hozzájárulás:

- A bemutatott implementáció az első, amely az MCF probléma vonatkozásában alkalmazza Goldberg *partial augment-relabel* ötletét [12]. Ezzel az algoritmus futásidejét általában 30-40%-kal sikerült csökkenti az eredeti push-relabel módszerhez képest.
- A reziduális hálózatot úgy reprezentáljuk, ahogy az SSP és CAS algoritmusban.
- A skálázási faktort 16-ra állítjuk.
- Hatékonyan alkalmazunk több korábban publikált heurisztikát: *potential refinement*, *global update* és *push-look-ahead*.

– **Network simplex (NS):** az általános szimplex módszer egy speciális változata (hálózati szimplex), amely az elemi műveleteket közvetlenül a gráfon hajtja végre. Az MCF problémát lineáris programozási (LP) feladatként tekintve a változók a gráf éleinek, az LP bázisok pedig feszítőfáknak felelnek meg.

Hozzájárulás:

- Az XTI módszert [8] alkalmazzuk a megengedett bázismegoldások reprezentálására (egy feszítőfát tárolunk különböző indexekkel). Ez sokkal hatékonyabbnak bizonyult, mint a népszerűbb ATI módszer.

- Az XTI módszert tovább javítottuk azáltal, hogy tárolunk még egy indexet, amely lehetővé teszi az adatszerkezet gyorsabb frissítését.
- Egy speciális gráfreprezentációt használunk, amelyben egy csúcs kimenő és bejövő éleinek listáit nem tároljuk.
- Kifejlesztettünk egy egyszerű heurisztikát a mesterséges inicializálási technika alapján, amely gyorsabbá teszi az első néhány iterációt.
- Hatékonyabb képleteket dolgoztunk ki két elterjedt pivotálási szabály fő paramétereinek beállításához. A *block search* és *candidate list* szabályok esetén a gráf éleiből alkotott blokkok vagy listák méretét m helyett $\lfloor\sqrt{m}\rfloor$ értékkel arányosan választjuk meg (ahol m a gráf éleinek száma). Ez sok esetben jelentősen csökkenti az algoritmus futásidejét (akár 40-50%-kal).
- Egy új pivotálási szabályt (*altering candidate list*) is kidolgoztunk, amely a gyakorlatban az egyik leghatékonyabbnak bizonyult.

2.2. Mérési eredmények

Átfogó tapasztalati elemzést végeztünk annak érdekében, hogy az implementált algoritmusokat összehasonlítsuk egymással és más megoldóprogramokkal: CS2, LEDA, MCFZIB, CPLEX, az MCFSimplex primál és duál változata, RelaxIV és PDNET. Ezek közül néhány (CS2, MCFZIB és RelaxIV) régóta viszonyítási pontként szolgál a szakirodalomban. A mérésekhez használt hálózatokat részben standard generátorokkal (NETGEN, GRIDGEN, GOTO és GRIDGRAPH), részben pedig valós problémák alapján generáltuk.

Eredmények és konklúziók

– A bemutatott tanulmány a korábbi elemzéseknél átfogóbb: az algoritmusoknak és az MCF feladat példányainak egyaránt egy szélesebb és változatosabb halmazát vizsgáljuk, köztük olyan hálózatokat is, amelyeknek több millió csúcsa és éle van.

– Általában a költségskálázó és primál hálózati szimplex algoritmusok a leggyorsabbak és legrobusztusabbak.

– Olyan hálózatok esetén, amelyekre az optimális folyamokat nem szükséges sok útra szétbontatni, az útjavító módszerek (SSP és CAS) a leghatékonyabbak.

– A szerző NS kódja lényegesen gyorsabbnak bizonyult, mint a módszer más implementációi: MCFZIB, MCFSimplex és CPLEX. Gyakran egy nagyságrenddel hatékonyabb.

– Viszonylag kicsi gráfokon (néhány tízezer csúcsig) általában az NS kód a leggyorsabb az összes vizsgált algoritmus közül.

– A költségskálázó algoritmusok ugyanakkor felülmúlják a hálózati szimplex módszert a nagy és ritka gráfokon, mivel a futásidejük a csúcsok számának függvényében jobb aszimptotikus viselkedést mutat.

– A szerző COS kódja hasonló teljesítményt ér el vagy kissé lassabb, mint a CS2, az algoritmus eredeti szerzője által készített rendkívül hatékony implementáció. Mindkét kód gyorsabb és stabilabb, mint a LEDA, amely szintén a költségskálázó módszert valósítja meg.

– Az elemzésben vizsgált további három megoldóprogram (az MCFSimplex duál változata, RelaxIV és PDNET) kevésbé robusztus. Gyakran nagyságrendekkel lassabbak, mint a primál hálózati szimplex és a költségskálázó algoritmusok, de bizonyos esetekben a RelaxIV kód nagyon hatékonynak bizonyult (pl. NETGEN hálózatokon).

3. Legnagyobb közös részgráfok

Az értekezésben vizsgált másik feladat két irányítatlan, csúcs- és élcímkézett gráf legnagyobb közös részgráfjának meghatározása. Ez egy klasszikus NP-nehéz optimalizálási probléma, amelynek alapvető fontosságú alkalmazásai vannak számos különböző területen, például kémiai és bioinformatika, mintafelismerés és számítógépes látás [9, 11].

A dolgozatban ezt a problémát kémiai alkalmazások szempontjából vizsgáljuk, amelyekben két molekula szerkezeti hasonlóságát gyakran a legnagyobb közös részgráfjuk segítségével jellemzik [10]. Egy molekula tipikus modellje egy irányítatlan címkézett gráf, amelynek csúcsai atomokat, élei pedig kémiai kötéseket reprezentálnak.

Ennek a feladatnak két elterjedt definíciója létezik: *maximum common induced subgraph* (MCIS) és *maximum common edge subgraph* (MCES). Munkánk során a kémiai informatikában relevánsabb MCES változatot vizsgáltuk, amelyik nem követeli meg, hogy a közös részgráfok feszítettek legyenek.

3.1. Implementált algoritmusok és heurisztikák

A szerző és Englert Péter egy közös munka keretében hatékony heurisztikákat fejlesztett ki az MCES feladathoz [1]. Kutatómunkájukat a ChemAxon-nál, egy kémiai és biológia alkalmazásokat és szolgáltatásokat fejlesztő szoftvercégnél végezték. A bemutatott algoritmusokat a ChemAxon több termékébe integrálták, amelyek vezető nemzetközi gyógyszercégek alkalmazásában állnak. Akadémiai kutatás és tesztelés céljából ezek a szoftverek ingyenesen hozzáférhetők a `https://chemaxon.com` weboldalon.

Az implementált algoritmusok az alábbiakban összefoglalt két ismert módszeren alapulnak.

– **Klikk-alapú algoritmus (clique-based algorithm, CB):** egy népszerű megközelítési mód, amely az MCIS/MCES feladatot visszavezeti a maximális klikk problémára egy, az eredeti gráfokból származtatott direktszorzat-gráf segítségével [16]. A bemutatott implementáció maximális klikkek keresésére Grosso–Locatelli–Pullan [13] heurisztikus algoritmusát alkalmazza.

– **Build-up algoritmus (BU):** Kawabata [14] által kidolgozott mohó heurisztikus algoritmus, amely közös részgráfok egy rendezett listájával dolgozik és fokozatosan bővíti azokat. Minden iterációban a közös részgráfokat kiterjeszti az összes lehetséges módon, és az így kapott részgráfok közül a legjobbakat tartja meg egy alkalmas értékelő függvény szerint.

Különböző heurisztikákat és implementációs javításokat alkalmazunk annak érdekében, hogy növeljük ezen algoritmusok pontosságát és sebességét, valamint az eredmények kémiai relevanciáját. Az alábbiakban ezeket a módszereket soroljuk fel.

– **A direktszorzat-gráf reprezentációja.** A CB algoritmusban a rendkívül nagy és sűrű direktszorzat-gráf helyett annak komplementerét tároljuk (szomszédsági listás ábrázolással), ami nagy mértékben javítja a módszer memóriaigényét és sebességét egyaránt. A reprezentáció mérete és felépítésének ideje $O(m^4)$ helyett $O(m^3)$, ahol m a két gráf élszámának maximuma.

– **Korai terminálás.** A CB algoritmusban a klikk-keresést befejezzük olyankor, amikor már biztosan megtaláltuk az optimális megoldást. Sok esetben biztosíthatunk ilyen garanciát azáltal, hogy kiszámolunk egy, a legnagyobb közös részgráf méretére vonatkozó felső korlátot.

– **Összefüggőségi heurisztika.** Az MCES algoritmusok egy fontos heurisztikája, hogy előnyben részesítik azokat a csúcsokat és éleket, amelyek kapcsolódnak a kiterjesztendő közös részgráfhoz. Ezt az ötletet eredményesen alkalmazzuk mindkét algoritmusban: a CB módszer esetén 1-4%-kal, a BU módszer esetén pedig átlagosan 10-20%-kal növelte a megtalált közös részgráfok méretét.

– **ECFP-alapú heurisztika.** Kidolgoztunk egy új heurisztikát az úgynevezett *extended connectivity fingerprintek* (ECFP) [17] generálási algoritmusát felhasználva. Az ECFP leírókat széles körben alkalmazzák molekulagráfok hasonlósági vizsgálatához. A módszer lényege, hogy minden csúcshoz egész hash-kódok sorozatát rendeljük, amelyek az adott csúcs növekvő sugárral vett környezeteit reprezentálják. Az MCES algoritmusokban ezeket a hash-kódokat arra használjuk, hogy előnyben részesítsük a két gráf olyan csúcsainak és éleinek egymáshoz rendelését, amelyek nagy izomorf környezettel rendelkeznek. Ez a heurisztika mindkét algoritmus esetén további 1-3%-kal növeli az eredmények méretét.

– **Leképezés optimalizálása.** Sok alkalmazásban (pl. reakciók elemzése, molekulák térbeli illesztése) a közös részgráf méretén kívül olyan topológiai tulajdonságok is fontosak, amelyeket az inputgráfok megfelelő csúcsait és éleit egymáshoz rendelő leképezés határoz meg. Kifejlesztettünk egy hatékony általános megoldást ennek a leképezésnek az optimalizálására, amely egy heurisztikus értékelő függvényen és azon csúcsok azonosításán alapul, amelyek leképezése jelentős lehet az alkalmazások szempontjából. A mérési eredményeink azt is megmutatták, hogy ez a módszer gyakran az algoritmus pontosságát is javítja azáltal, hogy elősegíti egy nagyobb közös részgráf megtalálását.

– **Gyűrűk megtartása.** Bizonyos kémiai alkalmazásokban (pl. klaszterezés, molekulák illesztése) gyakori követelmény, hogy csak olyan közös részgráfokat vizsgáljunk, amelyekben a gyűrűk (körök) nem szakadnak meg. Ezért kidolgoztunk egy egyszerű módszert ezen használati esetek támogatására.

A felsorolt ötletek közül néhány szerepelt korábban az irodalomban: a direktszorzat-gráf javított reprezentációja, a korai terminálás és az összefüggőségi heurisztika. A többi a szerző tudomása szerint új módszer. Bár molekulagráfok vizsgálatára fejlesztettük ki őket, ezek az ötletek alkalmazhatók más területeken is, ahol irányítatlan címkézett gráfok legnagyobb közös részgráfját szeretnénk meghatározni.

3.2. Mérési eredmények

A bemutatott implementációkat alaposan elemeztük és összehasonlítottuk más megoldóprogramokkal: a KCOMBU-val és a nyílt forrású Indigo programcsomag megfelelő metódusával.

Eredmények és konklúziók

- Az elemzés során két tényező is nehézséget okozott: egyrészt ehhez a feladathoz nincsenek standardnak tekinthető tesztgráfok, másrészt a különböző algoritmusok a probléma különböző változatait oldják meg (MCIS/MCES, összefüggő/nem összefüggő stb.).
- A méréseink igazolták, hogy a kifejlesztett heurisztikák jelentősen javítják az implementált algoritmusokat, az eredmények mérete és a futásidő szempontjából egyaránt.
- A CB algoritmus egyértelműen felülmúlja a BU módszert.
- A CB és a BU algoritmus is lényegesen pontosabb eredményeket ad és gyorsabb, mint a vizsgálatokba bevont másik két megoldóprogram, a KCOMBU és az Indigo.

4. A LEMON programkönyvtár

A LEMON [15] egy nyílt forrású C++ programkönyvtár, amely gráfoptimalizálási problémákhoz kapcsolódó különféle adatszerkezetek és algoritmusok hatékony implementációját tartalmazza. Magában foglalja a minimális költségű folyam feladat megoldására bemutatott algoritmusokat is, ezáltal elősegítve azok alkalmazását és más algoritmusokkal való kombinálását, ami komplex optimalizálási problémák megoldásához elengedhetetlen.

A szerző a programkönyvtár egyik fő fejlesztője, Jüttner Alpárral, a LEMON projekt vezetőjével, valamint Dezső Balázzsal együtt. Számos algoritmust implementált minimális költségű folyamok, minimális átlagú körök és maximális klikkek kereséséhez, továbbá részt vett a könyvtár alapvető funkcióinak fejlesztéséhez kapcsolódó tervezés, megvalósítás, felülvizsgálat és dokumentálás feladataiban is. Ezért az értekezésben a [4, 5] cikkek alapján röviden bemutatásra kerülnek a LEMON könyvtár tervezési elvei és koncepciói is.

Alapos elemzések igazolták, hogy a LEMON által nyújtott adatszerkezetek és algoritmusok tipikusan hatékonyabbak, mint más programkönyvtárak, a Boost Graph Library (BGL) és a LEDA megfelelő eszközei. A LEMON népszerűsége és különböző kutatási-fejlesztési projektekben való széleskörű alkalmazása kapcsán nélkülözhetetlennek bizonyult a könyvtár által

nyújtott gráfoptimalizálási algoritmusok széles választéka, különösen a minimális költségű folyam feladatra adott megoldási módszerek.

Az értekezés alapjául szolgáló közlemények

[1] P. Englert and P. Kovács. Efficient heuristics for maximum common substructure search. *J. Chem. Inf. Model.*, 55:941–955, 2015. DOI: 10.1021/acs.jcim.5b00036. IF: **3.657**.

[2] P. Kovács. Minimum-cost flow algorithms: an experimental evaluation. *Optim. Method Softw.*, 30:94–127, 2015. DOI: 10.1080/10556788.2014.895828. IF: **0.841**.

[3] Z. Király and P. Kovács. Efficient implementations of minimum-cost flow algorithms. *Acta Univ. Sapientiae, Inform.*, 4:67–118, 2012.

[4] B. Dezső, A. Jüttner, and P. Kovács. LEMON – an open source C++ graph template library. *Electron. Notes Theor. Comput. Sci.*, 264:23–45, 2011. DOI: 10.1016/j.entcs.2011.06.003.

[5] B. Dezső, A. Jüttner, and P. Kovács. LEMON – an open source C++ graph template library. In *Proc. 2nd Workshop on Generative Technologies*, WGT 2010, pages 3–13, Paphos, Cyprus, 2010.

[6] Z. Király and P. Kovács. An experimental study of minimum cost flow algorithms. In *Proc. 8th International Conference on Applied Informatics*, ICAI 2010, Vol. 2. Pages 227–235, Eger, Hungary, 2010.

Hivatkozások

[7] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Englewood Cliffs, NJ, USA, 1993. ISBN: 978-0136175490.

[8] R. S. Barr, F. Glover, and D. Klingman. Enhancements to spanning tree labelling procedures for network optimization. *INFOR*, 17:16–34, 1979.

[9] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *Int. J. Pattern Recognit. Artif. Intell.*, 18:265–298, 2004.

[10] E. Duesbury, J. D. Holliday, and P. Willett. Comparison of maximum common subgraph isomorphism algorithms for the alignment of 2D chemical structures. *ChemMedChem*, 13:588–598, 2018.

[11] H.-C. Ehrlich and M. Rarey. Maximum common subgraph isomorphism algorithms and their applications in molecular science: A review. *WIREs Comput. Mol. Sci.*, 1:68–79, 2011.

[12] A. V. Goldberg. The partial augment-relabel algorithm for the maximum flow problem. In *Proc. 16th Annual European Symposium on Algorithms*, ESA '08, pages 466–477, 2008.

[13] A. Grosso, M. Locatelli, and W. Pullan. Simple ingredients leading to very efficient heuristics for the maximum clique problem. *J. Heuristics*, 14:587–612, 2008.

[14] T. Kawabata. Build-up algorithm for atomic correspondence between chemical structures. *J. Chem. Inf. Model.*, 51:1775–1787, 2011.

[15] LEMON – Library for Efficient Modeling and Optimization in Networks. https://lemon.cs.elte.hu.

[16] J. W. Raymond, E. J. Gardiner, and P. Willett. RASCAL: Calculation of graph similarity using maximum common edge subgraphs. *Comput. J.*, 45:631–644, 2002.

[17] D. Rogers and M. Hahn. Extended-connectivity fingerprints. *J. Chem. Inf. Model.*, 50:742–754, 2010.

[18] A. Sifaleras. Minimum cost network flows: Problems, algorithms, and software. *Yugosl. J. Oper. Res.*, 23:3–17, 2013.